

## AN EFFICIENT TIMER MANAGEMENT SYSTEM

### TECHNICAL FIELD

5  
10  
15  
20  
25  
30  
35  
40  
45  
50  
55  
60  
65  
70  
75  
80  
85  
90  
95  
100  
105  
110  
115  
120  
125  
130  
135  
140  
145  
150  
155  
160  
165  
170  
175  
180  
185  
190  
195  
200  
205  
210  
215  
220  
225  
230  
235  
240  
245  
250  
255  
260  
265  
270  
275  
280  
285  
290  
295  
300  
305  
310  
315  
320  
325  
330  
335  
340  
345  
350  
355  
360  
365  
370  
375  
380  
385  
390  
395  
400  
405  
410  
415  
420  
425  
430  
435  
440  
445  
450  
455  
460  
465  
470  
475  
480  
485  
490  
495  
500  
505  
510  
515  
520  
525  
530  
535  
540  
545  
550  
555  
560  
565  
570  
575  
580  
585  
590  
595  
600  
605  
610  
615  
620  
625  
630  
635  
640  
645  
650  
655  
660  
665  
670  
675  
680  
685  
690  
695  
700  
705  
710  
715  
720  
725  
730  
735  
740  
745  
750  
755  
760  
765  
770  
775  
780  
785  
790  
795  
800  
805  
810  
815  
820  
825  
830  
835  
840  
845  
850  
855  
860  
865  
870  
875  
880  
885  
890  
895  
900  
905  
910  
915  
920  
925  
930  
935  
940  
945  
950  
955  
960  
965  
970  
975  
980  
985  
990  
995

The present invention relates to the field of managing timers in data processing systems, and more particularly to a timer management system that is used in both a synchronous and asynchronous system where an asynchronous application is synchronously acting on the timer in an asynchronous system and that allows a timer to be reinitialized without allocating new system memory.

### BACKGROUND INFORMATION

10  
15  
20  
25  
30  
35  
40  
45  
50  
55  
60  
65  
70  
75  
80  
85  
90  
95  
100  
105  
110  
115  
120  
125  
130  
135  
140  
145  
150  
155  
160  
165  
170  
175  
180  
185  
190  
195  
200  
205  
210  
215  
220  
225  
230  
235  
240  
245  
250  
255  
260  
265  
270  
275  
280  
285  
290  
295  
300  
305  
310  
315  
320  
325  
330  
335  
340  
345  
350  
355  
360  
365  
370  
375  
380  
385  
390  
395  
400  
405  
410  
415  
420  
425  
430  
435  
440  
445  
450  
455  
460  
465  
470  
475  
480  
485  
490  
495  
500  
505  
510  
515  
520  
525  
530  
535  
540  
545  
550  
555  
560  
565  
570  
575  
580  
585  
590  
595  
600  
605  
610  
615  
620  
625  
630  
635  
640  
645  
650  
655  
660  
665  
670  
675  
680  
685  
690  
695  
700  
705  
710  
715  
720  
725  
730  
735  
740  
745  
750  
755  
760  
765  
770  
775  
780  
785  
790  
795  
800  
805  
810  
815  
820  
825  
830  
835  
840  
845  
850  
855  
860  
865  
870  
875  
880  
885  
890  
895  
900  
905  
910  
915  
920  
925  
930  
935  
940  
945  
950  
955  
960  
965  
970  
975  
980  
985  
990  
995

A timer is a device which can be set to furnish an interrupt or a timeout indication at a specific time instant or after a selected time interval. Timers are required in data processing systems in which typical protocols require that a very large number of simultaneously occurring tasks or events be supervised to detect whether they occurred within predetermined delays. For example, a start operation may be sent by the application to start the timer in order to supervise a corresponding event. When the supervision of an event has to be interrupted for different reasons, a stop operation may be generated by the corresponding application. After a while, the supervision of the corresponding event may be requested to start again, in which case a start operation may be generated by the application. While the timer associated with an event is still running, the application may request a restart operation in order to delay the timing of the corresponding event.

ins A. A real-time operating system (RTOS), e.g., VxWorks™, OSE, Lynx, within a data processing system is basic software for the tool management of the hardware and the software of the system and provides hardware resource management functions, e.g., memory management, task management, data input/output management, and a user interface for handling a screen display and the manipulation of a mouse. An operating system may further include a module such as a timer management program, i.e., timer management system, for managing a plurality of timers that are started, stopped, idled, etc. by an application program in a data processing system. Application programs, e.g., word processing software, database software, software for calculation for tables, reside on top of the OS in the topmost layer of a hierarchial software arrangement.

Prior art timer management systems are used in either synchronous, i.e., single task, or asynchronous, i.e., multi-task, data processing systems. In a synchronous system, when the timer expires, the application, i.e., user of the timer, is notified by a message commonly referred to as a timer message. In an asynchronous system, when the timer expires, the timer message may be stored on queue before being sent to the application, i.e., user of the timer. If the application stops the timer prior to receiving a timer message, the timer moves back to the idle state. However, some operating systems do not allow timer messages to be removed from the queue when the application performs an operation on the expired timer prior to receiving the timer message. Subsequently, a special state-variable in the timer message denotes the fact that it is an illegal time-out message. In a synchronous system, this problem does not occur.

It would therefore be desirable to develop a timer management system that is used in both a synchronous and asynchronous system where an asynchronous application is synchronously acting on the timer in an asynchronous system. That is, it would therefore be desirable to develop a

function to filter these illegal time-out messages that are transparent to the application. It would further be desirable to allow a timer to be reinitialized without allocating new system memory.

00000000 00000000 00000000 00000000

**SUMMARY**

5 The problems outlined above may at least in part be solved in some embodiments by a handle function that allows an asynchronous application in a multi-task system, i.e., asynchronous system, to synchronously act on the timer. That is, when a timer in an asynchronous system times-out, the handle function filters the illegal time-out messages by allowing the asynchronous application to synchronously act on the timer. In another embodiment of the present invention, a timer may be reinitialized from the same allocated block of memory used to create the timer. In another embodiment of the present invention, a time-out message may be sent using the same allocated block of memory used to create the timer.

10 In one embodiment, a timer management system for managing timers in both a synchronous and asynchronous system comprises an application program interface (API) for providing a set of synchronous functions allowing an application to functionally operate on the timer. The timer management system further comprises a timer database for storing timer-related information. Furthermore, the timer management system comprises a timer services for detecting the expiring of the timer. A handle function of the timer services allows the asynchronous application, i.e., application in an asynchronous system, to synchronously act on the timer. When the timer expires, the handle function of the timer services allows the asynchronous application to act on the expired timer without incurring illegal time-out messages.

The foregoing has outlined rather broadly the features and technical advantages of the present invention in order that the detailed description of the invention that follows may be better understood. Additional features and advantages of the invention will be described hereinafter which form the subject of the claims of the invention.

5

#### **BRIEF DESCRIPTION OF THE DRAWINGS**

A better understanding of the present invention can be obtained when the following detailed description is considered in conjunction with the following drawings, in which:

Figure 1 illustrates a data processing system configured in accordance with the present invention;

Figure 2 illustrates an embodiment of a timer management system;

Figure 3 is a state machine which illustrates a timer management system of the present invention where an asynchronous application is synchronously acting on the timer in an asynchronous system; and

Figure 4 illustrates an embodiment of a data structure of a time-out message.

**DETAILED DESCRIPTION**

5 The present invention comprises a timer management system and method for managing  
timers in both a synchronous and asynchronous system. In one embodiment of the present invention,  
a timer management system comprises an application program interface (API) for providing a set  
of synchronous functions allowing an application to functionally operate on the timer. The timer  
management system further comprises a timer database for storing timer-related information.  
Furthermore, the timer management system comprises a timer services for detecting the expiring of  
10 the timer. A handle function of the timer services allows an asynchronous application, i.e.,  
application in a multi-task system, to synchronously act on the timer. That is, when a timer in an  
asynchronous system times-out, the handle function allows the asynchronous application to act on  
the expired timer without incurring an illegal time-out message. In another embodiment of the  
present invention, a timer may be reinitialized from the same allocated block of memory used to  
15 create the timer. In another embodiment of the present invention, a time-out message may be sent  
using the same allocated block of memory used to create the timer.

Figure 1 - Computer System

20 *ms* Figure 1 illustrates a typical hardware configuration of data processing system 13 which is  
representative of a hardware environment for practicing the present invention. Data processing  
system 13 has a central processing unit (CPU) 10, such as a conventional microprocessor, coupled  
to various other components by system bus 12. A real-time operating system 40, e.g., VxWorks™,

5  
10  
15  
OSE, Lynx, runs on CPU 10 and provides control and coordinates the function of the various components of Figure 1. As stated in the Background Information section, a timer management program, i.e., timer management system, may reside in a module within the operating system 40. In another embodiment, an application 42, e.g., timer management system, may run in conjunction with operating system 40 and provide output calls to operating system 40 which implements the various functions to be performed by application 42. Read only memory (ROM) 16 is coupled to system bus 12 and includes a basic input/output system ("BIOS") that controls certain basic functions of data processing system 13. Random access memory (RAM) 14, I/O adapter 18, and communications adapter 34 are also coupled to system bus 12. It is noted that software components including operating system 40 and application 42 are loaded into RAM 14 which is the computer system's main memory. I/O adapter 18 may be a small computer system interface ("SCSI") adapter that communicates with disk units 20 and tape drives 40. Communications adapter 34 interconnects bus 12 with an outside network enabling data processing system 13 to communication with other such systems. Input/Output devices are also connected to system bus 12 via a user interface adapter 22 and a display adapter 36. A display monitor 38 is connected to system bus 12 by display adapter 36. In this manner, a user is capable of inputting to system 13 through a keyboard 24 or a mouse 26 and receiving output from system 13 via display 38.

20  
Preferred implementations of the invention include implementations as a computer system programmed to execute the method or methods described herein, and as a computer program product. According to the computer system implementations, sets of instructions for executing the method or methods are resident in the random access memory 14 of one or more computer systems configured generally as described above. Until required by the computer system, the set of

instructions may be stored as a computer program product in another computer memory, for example, in disk drive 20 (which may include a removable memory such as an optical disk or floppy disk for eventual use in disk drive 20). Furthermore, the computer program product can also be stored at another computer and transmitted when desired to the user's work station by a network or by an external network such as the Internet. One skilled in the art would appreciate that the physical storage of the sets of instructions physically changes the medium upon which it is stored so that the medium carries computer readable information. The change may be electrical, magnetic, chemical or some other physical change.

#### Figure 2 - Timer Management System

Figure 2 illustrates an embodiment of a timer management system 200. As stated in the Background Information section, an operating system may include a module that comprises timer management system 200. The operating system typically resides within the kernel of data processing system 13. In another embodiment, timer management system 200 may be an application program residing in the topmost layer of a typical hierarchical software arrangement independent of the operating system.

Timer management system 200 comprises an application program interface (API) 220, a timer database 230 and a timer services 240. API 220 allows software application(s) 210 to functionally operate timers, e.g., start, stop, restart, delete, in data processing system 13. It is noted that for simplicity, software applications 210 that use timer management system services may be referred to as "users." API 220 provides a set of synchronous functions for creating, starting,



stopping and deleting a timer. In one embodiment, API 220 may be a dynamic-link library (DLL) file. A DLL file is one that contains one or more functions that are compiled, linked and stored separately from application processes that use them, and exports code that may be shared among applications. Furthermore, the set of synchronous functions provided by API 220 allows user application(s) 210 to access timer database 230. Timer database 230 may be a repository for timer-related information, e.g., timer records for created timers. It is noted that timer database 230 may be structured in any form, e.g., link list, interval list.

Timer services 240, an OS task, detects the expiration of timers corresponding to each application 210 that has initiated a timer via API 220. Timer services 240 maintains timer database 230 as well as scans timer database 230 for expired timers. The scan of timer database 230 may be performed at regular time intervals. If the timer has expired, timer services 240 takes appropriate action depending on whether the timer management system is implemented in a single-task system or a multi-task system. In a single-task system, i.e., synchronous system, as will be described in further detail below, timer services 240 calls a time-out function to send a time-out message to user application 210. In a multi-task system, i.e., asynchronous system, as will be described in further detail below, timer services 240 sends a time-out message to the application's task if the timer has expired. The time-out message sent is the same block of memory as the block of memory allocated when the timer was created as will be further described in conjunction with Figure 3. As stated in the Background Information section, the timer message may be stored on queue before being sent to application 210 in a multi-task system. If application 210 stops the timer prior to receiving a timer message, the timer moves back to the idle state. However, some operating systems do not allow timer messages to be removed from the queue when application 210 performs an operation on the

expired timer prior to receiving the timer message. Subsequently, a special state-variable in the timer message denotes the fact that it is an illegal time-out message. A detailed explanation of timer management system 200 avoiding illegal time-out messages by allowing an asynchronous application 210 to synchronously act on the timer in an asynchronous system is provided below. Furthermore, a detailed explanation of timer management system 200 allowing a timer to be reinitialized without allocating new system memory is provided below.

Figure 3 - State Machine Illustration of Timer management system

Figure 3 illustrates an embodiment of the present invention of a state machine 300 which may be used in the operation of timer management system 200. State machine 300 depicts a timer management system 200 that manages timers for both a single task system, i.e., synchronous system, and a multi-task system, i.e., asynchronous system, concurrently. State machine 300 includes a single-task state machine 301 and a multi-task state machine 302. The states of the single-task state machine 301 comprise states 303, 304 and 305. The states of the multi-task state machine 302 comprise states 306, 307, 308 and 309.

Referring to Figure 3, state machine 300 remains in state 303, non-existent ("NE"), until a timer is created. That is, state 303 represents a state in which a timer is non-existent. When a timer is created ("C"), state machine 300 transitions from state 303 to state 304. State 304 represents a state in which a timer is created and ready to be initialized. That is, the timer is idle ("I"). In one embodiment, timer services 240 creates a timer out of a block of memory. The block of memory allocated for creating the timer may then be stored in timer database 230. A description of the block

A of memory allocated for creating the timer is provided below in connection with the data structure of the time-out message.

If the timer is deleted ("D") while in state 304, then state machine 300 transitions from state 304 to state 303. If the timer is activated ("A") while at state 304, then state machine 300 transitions from state 304 to state 305. State 305 represents a state in which the timer that was previously created is running ("R"). If the timer is deleted while at state 305, then state machine transitions from state 305 to state 303. If the timer that is running is stopped ("O"), then state machine 300 transitions from state 305 to state 304. If the timer is not deleted at state 304 but instead is activated, the timer will be reinitialized, i.e., restarted, without allocating new system memory, i.e., using the same system memory. If the timer at state 305 expires, then a time-out message ("Ts") is notified synchronously to application 210 and state machine 300 transitions from state 305 to state 303.

As stated above, a time-out message sent is the same block of memory as the block of memory allocated when the timer was created. The reason is that the procedure that creates the timer creates extra memory space for use by timer services 240 which is not seen by application 210. A drawing of an embodiment of a data structure 400 of the time-out message is illustrated in Figure 4. A time-out message may comprise at least two distinct parts. An application part 410 may comprise information related to application 210. In a single-task system, the application part 410 may comprise the time-out function and a context field which are owned by application 210. In a multi-task system, the application part 410 may comprise the same context field as well as information necessary to identify application 210, e.g., queue to send the message, application task identification. A timer services part 420 may comprise some additional fields, e.g., pointers to

A memory addresses of timers stored in timer database 230. The timer services part 420 is not seen by application 210.

Inst A 6 Referring to Figure 3, application 210 may erroneously believe that the timer is operating at a state in which it is not. For example, application 210 may mistakenly believe that the timer is running when the timer is in actuality idle at state 304. Application 210 may then stop the timer at the idle state which is represented by the arched line at state 304. Furthermore, application 210 may mistakenly believe that the timer is idle when the timer is in actuality running at state 305. Application 210 may then activate the timer at the running state which is represented by the arched line at state 305.

10 In multi-task system, i.e., asynchronous system, a time-out message is sent to the application's task. As stated in the Background Information section, when the timer expires in an asynchronous system, the timer message may be stored on queue before being sent to application 210. If application 210 stops the timer prior to receiving a timer message, the timer moves back to the idle state. However, some operating systems do not allow timer messages to be removed from the queue when application 210 performs an operation on the expired timer prior to receiving the timer message. Subsequently, a special state-variable in the timer message denotes the fact that it is an illegal time-out message. A description of a process that filters these illegal time-out messages that are transparent to user application 210 is provided below.

20 In an asynchronous system, when the timer at state 305 expires, a time-out message ("Tm") is queued. In one embodiment, the time-out messages, Tm, may be stored in a system queue attached to application 210. It is noted that the queue storing the time-out messages, Tm, may be

located anywhere in system memory. When the timer at state 305 expires in an asynchronous system, state machine 300 transitions from state 305 to state 306. State 306 is the expired state ("E") where the timer is not running and the time-out message, Tm, is queued. As soon as application 210 receives the time-out message, Tm, a handle function ("H") of timer services 240 transparent to application 210 allows application 210 in an asynchronous system to synchronously act on the timer. That is, the handle function transfers state machine 300 from a state in the multi-task state machine 302 to a state in the single-task state machine 301. For example, if application 210 receives the time-out message, Tm, when state machine is at state 306, then the handle function transfers state machine 300 from state 306 in the multi-task state machine 302 to the idle state 304 in the single-task state machine 301.

If the asynchronous application 210, i.e., application of an asynchronous system, has not received the time-out message, Tm, stored in a queue, e.g., a system queue attached to application 210, then asynchronous application 210 may not know that the timer has expired. Asynchronous application 210 may stop the timer erroneously believing that the timer is running. Asynchronous application 210 may then believe the timer is in the idle state as represented by a transition from state 306 to state 307. State 307 is the state ("IQ") in which the timer is in the idle state and the time-out message, Tm, is still queued. When the time-out message is dequeued, the handle function transparent to asynchronous application 210 transfers state machine 300 from state 307 in the multi-task state machine 302 to the idle state 304 in the single-task state machine 301. If while at state 306, asynchronous application 210 deletes the timer, then state machine 300 transitions from state 306 to state 308. State 308 is the state ("DQ") in which the timer is deleted and the time-out message, Tm, is still queued. When the time-out message is dequeued, the handle function

transparent to asynchronous application 210 transfers state machine 300 from state 308 in the multi-task state machine 302 to the non-existent state 303 in the single-task state machine 301. If while at state 306, asynchronous application 210 activates the timer, then state machine 300 transitions from state 306 to state 309. State 309 is the state ("RQ") in which the timer is in the running state and the time-out message, Tm, is still queued. When the time-out message is dequeued, the handle function transparent to asynchronous application 210 transfers state machine 300 from state 309 in the multi-task state machine 302 to the running state 305 in the single-task state machine 301.

5  
10  
15  
20  
30  
40  
50  
60  
70  
80  
90  
100  
110  
120  
130  
140  
150  
160  
170  
180  
190  
200  
210  
220  
230  
240  
250  
260  
270  
280  
290  
300  
310  
320  
330  
340  
350  
360  
370  
380  
390  
400  
410  
420  
430  
440  
450  
460  
470  
480  
490  
500  
510  
520  
530  
540  
550  
560  
570  
580  
590  
600  
610  
620  
630  
640  
650  
660  
670  
680  
690  
700  
710  
720  
730  
740  
750  
760  
770  
780  
790  
800  
810  
820  
830  
840  
850  
860  
870  
880  
890  
900  
910  
920  
930  
940  
950  
960  
970  
980  
990  
1000  
1010  
1020  
1030  
1040  
1050  
1060  
1070  
1080  
1090  
1100  
1110  
1120  
1130  
1140  
1150  
1160  
1170  
1180  
1190  
1200  
1210  
1220  
1230  
1240  
1250  
1260  
1270  
1280  
1290  
1300  
1310  
1320  
1330  
1340  
1350  
1360  
1370  
1380  
1390  
1400  
1410  
1420  
1430  
1440  
1450  
1460  
1470  
1480  
1490  
1500  
1510  
1520  
1530  
1540  
1550  
1560  
1570  
1580  
1590  
1600  
1610  
1620  
1630  
1640  
1650  
1660  
1670  
1680  
1690  
1700  
1710  
1720  
1730  
1740  
1750  
1760  
1770  
1780  
1790  
1800  
1810  
1820  
1830  
1840  
1850  
1860  
1870  
1880  
1890  
1900  
1910  
1920  
1930  
1940  
1950  
1960  
1970  
1980  
1990  
2000  
2010  
2020  
2030  
2040  
2050  
2060  
2070  
2080  
2090  
2100  
2110  
2120  
2130  
2140  
2150  
2160  
2170  
2180  
2190  
2200  
2210  
2220  
2230  
2240  
2250  
2260  
2270  
2280  
2290  
2300  
2310  
2320  
2330  
2340  
2350  
2360  
2370  
2380  
2390  
2400  
2410  
2420  
2430  
2440  
2450  
2460  
2470  
2480  
2490  
2500  
2510  
2520  
2530  
2540  
2550  
2560  
2570  
2580  
2590  
2600  
2610  
2620  
2630  
2640  
2650  
2660  
2670  
2680  
2690  
2700  
2710  
2720  
2730  
2740  
2750  
2760  
2770  
2780  
2790  
2800  
2810  
2820  
2830  
2840  
2850  
2860  
2870  
2880  
2890  
2900  
2910  
2920  
2930  
2940  
2950  
2960  
2970  
2980  
2990  
3000  
3010  
3020  
3030  
3040  
3050  
3060  
3070  
3080  
3090  
3100  
3110  
3120  
3130  
3140  
3150  
3160  
3170  
3180  
3190  
3200  
3210  
3220  
3230  
3240  
3250  
3260  
3270  
3280  
3290  
3300  
3310  
3320  
3330  
3340  
3350  
3360  
3370  
3380  
3390  
3400  
3410  
3420  
3430  
3440  
3450  
3460  
3470  
3480  
3490  
3500  
3510  
3520  
3530  
3540  
3550  
3560  
3570  
3580  
3590  
3600  
3610  
3620  
3630  
3640  
3650  
3660  
3670  
3680  
3690  
3700  
3710  
3720  
3730  
3740  
3750  
3760  
3770  
3780  
3790  
3800  
3810  
3820  
3830  
3840  
3850  
3860  
3870  
3880  
3890  
3900  
3910  
3920  
3930  
3940  
3950  
3960  
3970  
3980  
3990  
4000  
4010  
4020  
4030  
4040  
4050  
4060  
4070  
4080  
4090  
4100  
4110  
4120  
4130  
4140  
4150  
4160  
4170  
4180  
4190  
4200  
4210  
4220  
4230  
4240  
4250  
4260  
4270  
4280  
4290  
4300  
4310  
4320  
4330  
4340  
4350  
4360  
4370  
4380  
4390  
4400  
4410  
4420  
4430  
4440  
4450  
4460  
4470  
4480  
4490  
4500  
4510  
4520  
4530  
4540  
4550  
4560  
4570  
4580  
4590  
4600  
4610  
4620  
4630  
4640  
4650  
4660  
4670  
4680  
4690  
4700  
4710  
4720  
4730  
4740  
4750  
4760  
4770  
4780  
4790  
4800  
4810  
4820  
4830  
4840  
4850  
4860  
4870  
4880  
4890  
4900  
4910  
4920  
4930  
4940  
4950  
4960  
4970  
4980  
4990  
5000  
5010  
5020  
5030  
5040  
5050  
5060  
5070  
5080  
5090  
5100  
5110  
5120  
5130  
5140  
5150  
5160  
5170  
5180  
5190  
5200  
5210  
5220  
5230  
5240  
5250  
5260  
5270  
5280  
5290  
5300  
5310  
5320  
5330  
5340  
5350  
5360  
5370  
5380  
5390  
5400  
5410  
5420  
5430  
5440  
5450  
5460  
5470  
5480  
5490  
5500  
5510  
5520  
5530  
5540  
5550  
5560  
5570  
5580  
5590  
5600  
5610  
5620  
5630  
5640  
5650  
5660  
5670  
5680  
5690  
5700  
5710  
5720  
5730  
5740  
5750  
5760  
5770  
5780  
5790  
5800  
5810  
5820  
5830  
5840  
5850  
5860  
5870  
5880  
5890  
5900  
5910  
5920  
5930  
5940  
5950  
5960  
5970  
5980  
5990  
6000  
6010  
6020  
6030  
6040  
6050  
6060  
6070  
6080  
6090  
6100  
6110  
6120  
6130  
6140  
6150  
6160  
6170  
6180  
6190  
6200  
6210  
6220  
6230  
6240  
6250  
6260  
6270  
6280  
6290  
6300  
6310  
6320  
6330  
6340  
6350  
6360  
6370  
6380  
6390  
6400  
6410  
6420  
6430  
6440  
6450  
6460  
6470  
6480  
6490  
6500  
6510  
6520  
6530  
6540  
6550  
6560  
6570  
6580  
6590  
6600  
6610  
6620  
6630  
6640  
6650  
6660  
6670  
6680  
6690  
6700  
6710  
6720  
6730  
6740  
6750  
6760  
6770  
6780  
6790  
6800  
6810  
6820  
6830  
6840  
6850  
6860  
6870  
6880  
6890  
6900  
6910  
6920  
6930  
6940  
6950  
6960  
6970  
6980  
6990  
7000  
7010  
7020  
7030  
7040  
7050  
7060  
7070  
7080  
7090  
7100  
7110  
7120  
7130  
7140  
7150  
7160  
7170  
7180  
7190  
7200  
7210  
7220  
7230  
7240  
7250  
7260  
7270  
7280  
7290  
7300  
7310  
7320  
7330  
7340  
7350  
7360  
7370  
7380  
7390  
7400  
7410  
7420  
7430  
7440  
7450  
7460  
7470  
7480  
7490  
7500  
7510  
7520  
7530  
7540  
7550  
7560  
7570  
7580  
7590  
7600  
7610  
7620  
7630  
7640  
7650  
7660  
7670  
7680  
7690  
7700  
7710  
7720  
7730  
7740  
7750  
7760  
7770  
7780  
7790  
7800  
7810  
7820  
7830  
7840  
7850  
7860  
7870  
7880  
7890  
7900  
7910  
7920  
7930  
7940  
7950  
7960  
7970  
7980  
7990  
8000  
8010  
8020  
8030  
8040  
8050  
8060  
8070  
8080  
8090  
8100  
8110  
8120  
8130  
8140  
8150  
8160  
8170  
8180  
8190  
8200  
8210  
8220  
8230  
8240  
8250  
8260  
8270  
8280  
8290  
8300  
8310  
8320  
8330  
8340  
8350  
8360  
8370  
8380  
8390  
8400  
8410  
8420  
8430  
8440  
8450  
8460  
8470  
8480  
8490  
8500  
8510  
8520  
8530  
8540  
8550  
8560  
8570  
8580  
8590  
8600  
8610  
8620  
8630  
8640  
8650  
8660  
8670  
8680  
8690  
8700  
8710  
8720  
8730  
8740  
8750  
8760  
8770  
8780  
8790  
8800  
8810  
8820  
8830  
8840  
8850  
8860  
8870  
8880  
8890  
8900  
8910  
8920  
8930  
8940  
8950  
8960  
8970  
8980  
8990  
9000  
9010  
9020  
9030  
9040  
9050  
9060  
9070  
9080  
9090  
9100  
9110  
9120  
9130  
9140  
9150  
9160  
9170  
9180  
9190  
9200  
9210  
9220  
9230  
9240  
9250  
9260  
9270  
9280  
9290  
9300  
9310  
9320  
9330  
9340  
9350  
9360  
9370  
9380  
9390  
9400  
9410  
9420  
9430  
9440  
9450  
9460  
9470  
9480  
9490  
9500  
9510  
9520  
9530  
9540  
9550  
9560  
9570  
9580  
9590  
9600  
9610  
9620  
9630  
9640  
9650  
9660  
9670  
9680  
9690  
9700  
9710  
9720  
9730  
9740  
9750  
9760  
9770  
9780  
9790  
9800  
9810  
9820  
9830  
9840  
9850  
9860  
9870  
9880  
9890  
9900  
9910  
9920  
9930  
9940  
9950  
9960  
9970  
9980  
9990  
10000

If while at state 307, asynchronous application 210 deletes the timer, then state machine 300 transitions from state 307 to state 308. When the time-out message is dequeued, the handle function transfers state machine 300 from state 308 in the multi-task state machine 302 to the non-existent state 303 in the single-task state machine 301. If while at state 307, asynchronous application 210 activates the timer, then state machine 300 transitions from state 307 to state 309. When the time-out message is dequeued, the handle function transparent to asynchronous application 210 transfers state machine 300 from state 309 in the multi-task state machine 302 to the running state 305 in the single-task state machine 301. It is noted that when the timer is reinitialized, i.e., restarted, in state 309 the timer uses the same allocated system memory as the system memory allocated when the timer was created in state 304.

If while at state 309, asynchronous application 210 stops the timer, then state machine 300 transitions from state 309 to state 307. When the time-out message is dequeued, the handle function transparent to asynchronous application 210 transfers state machine 300 from state 307 in the multi-task state machine 302 to the idle state 304 in the single-task state machine 301. If while at

state 309, asynchronous application 210 deletes the timer, then state machine 300 transitions from state 309 to state 308. When the time-out message is dequeued, the handle function transparent to asynchronous application 210 transfers state machine 300 from state 308 in the multi-task state machine 302 to the non-existent state 303 in the single-task state machine 201.

5           As in the single-task state machine 301, asynchronous application 210 may erroneously  
believe that the timer is operating at a state in which it is not. For example, asynchronous  
application 210 may mistakenly believe that the timer is running when the timer is in actuality idle  
at state 307. Asynchronous application 210 may then stop the timer at state 307 which is represented  
by the arched line at state 307. Furthermore, asynchronous application 210 may mistakenly believe  
10           that the timer is idle when the timer is in actuality running at state 309. Asynchronous application  
210 may then activate the timer at the running state which is represented by the arched line at state  
309.

15           Although the timer management system and method of the present invention are described  
in connection with several embodiments, it is not intended to be limited to the specific forms set  
forth herein, but on the contrary, it is intended to cover such alternatives, modifications, and  
equivalents, as can be reasonably included within the spirit and scope of the invention as defined by  
the appended claims. It is noted that the headings are used only for organizational purposes and not  
meant to limit the scope of the description or claims.